



RECEIVED  
CENTRAL FAX CENTER  
JUN 28 2006

1500 K Street, NW  
Washington, DC 20005-1257  
202.220.4200  
Fax 202.220.4201

Fax Transmission

From: David A. Klein Date: June 28, 2006  
Direct Dial: 202.220.4209 Fax: 202.220.4201  
Client/Matter: 2207/8754 Total number of pages: 32  
(including cover)

Please deliver to:

Name	Company	Fax	Phone
Mail Stop Amendment	USPTO	571.273.8300	

Comments:

Application No. 09/536,452

Group Art Unit: 2183

Applicant: Ronny RONEN

Examiner: David J. HUISMAN

Filed: March 28, 2000

Confirmation No. 5160

For: METHOD AND APPARATUS FOR EXECUTING A 32-BIT APPLICATION BY  
CONFINING THE APPLICATION TO A 32-BIT ADDRESS SPACE SUBSET IN A  
64-BIT PROCESSOR

Please see Transmittal Form, Certificate of Transmission, Appeal Brief, and Evidence  
Appendix, attached.

☒ Original will not follow ☐ Original will follow by ☐ Regular Mail ☐ Overnight Delivery ☐ Hand Delivery

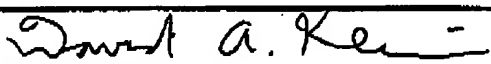
The information contained in this facsimile transmission, including any attachments, is subject to the attorney-client privilege, the attorney work product privilege or is confidential information intended only for the use of the named recipient. If the reader of this Notice is not the intended recipient or the employee or agent responsible for delivering this transmission to the intended recipient, you are hereby notified that any use, dissemination, distribution or copying of this communication is strictly prohibited. If you have received this transmission in error, please notify us immediately by telephone, so that we may arrange for its return or destruction at our cost. Thank you.

New York Washington, DC Silicon Valley www.kenyon.com

BEST AVAILABLE COPY

<b>TRANSMITTAL FORM</b>  <i>(to be used for all correspondence after initial filing)</i>	Application Number	09/536,452	<b>RECEIVED</b> <b>CENTRAL FAX CENTER</b> <b>JUN 28 2006</b>
	Filing Date	March 28, 2000	
	First Named Inventor	Ronny Ronen et al.	
	Art Unit	2183	
	Examiner Name	David J. Huisman	
Total Number of Pages in This Submission	32	Attorney Docket Number	02207/8754

ENCLOSURES (check all that apply)		
<input type="checkbox"/> Fee Transmittal Form <input type="checkbox"/> Fee Attached <input type="checkbox"/> Amendment <input type="checkbox"/> After Final <input type="checkbox"/> Affidavits/declaration(s) <input checked="" type="checkbox"/> Extension of Time Request <input type="checkbox"/> Express Abandonment Request <input type="checkbox"/> Information Disclosure Statement <input type="checkbox"/> Certified Copy of Priority Document(s) <input type="checkbox"/> Reply to Notice to File Corrected Application Papers <input type="checkbox"/> Reply to Missing Parts under 37 CFR 1.52 or 1.53	<input type="checkbox"/> Drawing(s) <input type="checkbox"/> Licensing-related Papers <input type="checkbox"/> Petition <input type="checkbox"/> Petition to Convert to a Provisional Application <input type="checkbox"/> Power of Attorney, Revocation Change of Correspondence Address <input type="checkbox"/> Terminal Disclaimer <input type="checkbox"/> Request for Refund <input type="checkbox"/> CD, Number of CD(s) _____ <input type="checkbox"/> Landscape Table on CD	<input type="checkbox"/> After Allowance Communication to TC <input checked="" type="checkbox"/> Appeal Communication to Board of Appeals and Interferences <input type="checkbox"/> Appeal Communication to TC (Appeal Notice, Brief, Reply Brief) <input type="checkbox"/> Proprietary Information <input type="checkbox"/> Status Letter <input checked="" type="checkbox"/> Other Enclosure(s) (please identify below): Evidence Appendix Fax Transmission Certificate of Transmission
<b>Remarks</b> Please charge the fee for filing a two-month Extension of Time (\$450) and for filing an Appeal Brief (\$500) to Deposit Account No. 11-0800.		

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT			
Firm	Kenyon & Kenyon LLP		
Signature			
Printed Name	David A. Klein		
Date	June 28, 2006	Reg. No.	46,835

CERTIFICATE OF TRANSMISSION/MAILING			
I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below.			
Signature			
Typed or printed name		Date	

JUN 28 2006

PATENT  
Atty. Docket No. 2207/8754

## IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No. : 09/536,452 Confirmation No. 5160  
Applicant : Ronny Ronen *et al.*  
Filed : March 28, 2000  
For : Method and Apparatus For Execcuting A 32-bit Application By  
Confining The Application To A 32-bit Address Space Subset In A  
64-bit Processor  
Group Art Unit : 2183  
Examiner : David J. Huisman  
Docket No. : 2207/8754  
Customer No. : 23838

Mail Stop Appeal Brief - Patents  
COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, VA 22313-1450

## APPEAL BRIEF UNDER 37 C.F.R. § 41.37

Sir:

This brief is in furtherance of the Notice of Appeal filed on February 28, 2006.

**I. REAL PARTY IN INTEREST**

Intel Corporation is the real party in interest for all issues related to this application by virtue of assignments recorded on June 2, 2000, at Reel 010823/Frame 0629.

**II. RELATED APPEALS AND INTERFERENCES**

There are no other appeals, interferences, or judicial proceedings known to appellants, appellants' legal representative, or assignee which may be related to, directly affect, or be directly affected by or have a bearing on the Board's decision in the pending appeal.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

### **III. STATUS OF CLAIMS**

Claims 1, 2, 4, 7-9, 11-15, 17, 18 and 22-28 are pending in this proceeding. Claims 3, 5, 6, 10, 16, and 19-21 were cancelled; claim 27 was objected to under 37 C.F.R. § 1.75(c) and rejected under 35 U.S.C. § 112, second paragraph; claims 1, 2, 4, 7-9, 11, 13-15, 17-18, 22, 25-26, and 28 were rejected under 35 U.S.C. §§ 102(b); claims 12, 23, and 24 were objected to as being dependent upon a rejected base claim, but indicated as being allowable if rewritten in independent form including all of the limitation of the base claim and any intervening claims. Claims 1, 2, 4, 7-9, 11, 13-15, 17-18, 22, 25-26, and 28 stand finally rejected and are the subject of this appeal.

### **IV. STATUS OF AMENDMENTS**

Subsequent to the Final Office Action dated October 28, 2005, appellants filed a Response Under 37 C.F.R. § 1.116 on January 30, 2006, but no claims were amended. In an Advisory Action dated February 16, 2006, the Examiner indicated that the request for reconsideration has been considered, but does not place the application in condition for allowance. Appellants filed an Amendment Under 37 C.F.R. § 41.33(a) on April 27, 2006 to correct a minor error in claim 18 set forth in the Advisory Action. The attached claims reflect their status as of the October 28, 2005 Final Office Action, but incorporate the amendment to claim 18 submitted April 27, 2006. According to PAIR, the Examiner initialed the Amendment filed April 27, 2006 on May 11, 2006. However, entry (or denial thereof) of the Amendment dated April 27, 2006 is not consequential to the issues on appeal.

### **V. SUMMARY OF CLAIMED SUBJECT MATTER**

The present invention relates to confining a software application executed on a microprocessor to a subset of the memory address space of the microprocessor. For example, confining a software application to a 32-bit address space subset on a 64-bit microprocessor.

Porting a 32-bit application into a 64-bit environment can result in errors when a 32-bit address is moved or is part of an operation within a 64-bit general purpose register (e.g., page 1, lines 10-11; page 3, lines 12-15). For example, if a 32-bit number is consigned to the low-order 32-bits in a 64-bit environment, incorrect content can be introduced when a wraparound error

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

causes a carry bit from the low-order 32 bits to be propagated into the high-order 32 bits, resulting in computational errors (*e.g.*, page 1, lines 11-16; page 3, lines 15-21). For example, when the 32-bit address is used to address a data item in memory in a 64-bit address space, the computation error can lead to a load or store from an incorrect address (*e.g.*, page 4, lines 7-9).

Referring to FIG. 1, a microprocessor 100 in accordance with embodiments of the present invention includes several control “flags” 112/114/116 (*e.g.*, control bits) and associated addressing mode control logic 110.

A first control flag may be the “address space control flag” 112. The address space control flag 112 may be set to indicate to the microprocessor 100 whether a software application uses 32-bit addresses (*e.g.*, page 2, lines 10-12; page 4, lines 12-15).

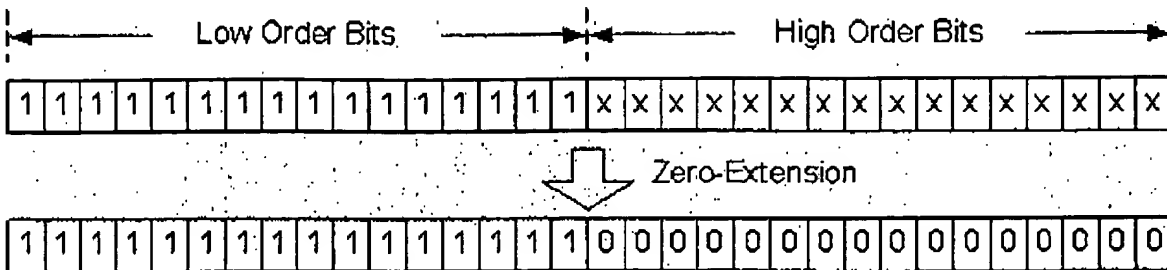
A second control flag may be the “address format control flag” 114. The address format control flag 114 specifies whether a 32-bit address is an unsigned address or a signed address (*e.g.*, page 2, lines 14-18). For example, a 32-bit application can operate in an *unsigned* 0 to 4 GB (gigabyte) virtual address space, or a *signed* -2 GB to +2 GB virtual address space (*e.g.*, page 2, line 25 to page 3, line 2).

A third control flag may be the “address fault control flag” 116. The address fault control flag 116 specifies whether the processor 100 should generate a “fault” when a generated address is not within a proper 32-bit address space subset (*e.g.*, page 2, lines 19-21).

Referring to FIG. 2, if the address space control flag 112 is set, the instruction execution core 130 truncates (240) the 64-bit generated address to 32 bits (*e.g.*, page 5, lines 8-9; page 8, lines 1-2). If the address format control flag indicates (250) that the 32-bit application uses an *unsigned* address space, the truncated address is then zero-extended (270) to 64 bits (*e.g.*, page 5, lines 12-14; page 5, line 22 to page 6, line 1; page 8, lines 5-6). As is commonly understood in the art, zero extension is performed by replacing the higher-order bits (*e.g.*, the high-order 32 bits) with zeros. For example, the following picture illustrates zero extending a 16-bit address to 32-bits:

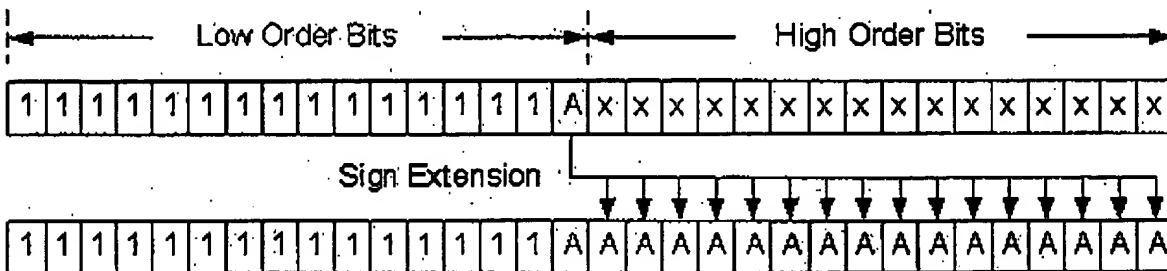
Appcal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754



where the content "x" is unimportant.

In comparison, if the address format control flag indicates (250) that the 32-bit application uses a *signed* address space, the truncated address is sign extended (260) to 64 bits (e.g., page 5, lines 9-12; page 6, lines 1-4; page 8, lines 2-5). As is commonly understood in the art, sign extension is performed by replacing the higher-order bits (e.g., the high-order 32 bits) with the value in the highest bit of the lower-order bits. For example, the following picture illustrates sign-extending a 16-bit address to 32-bits:



where "A" conveys the sign bit value.

If the highest bit of the lower-order bits is a zero, both zero extension and sign extension will produce the same result. However, zero extension can be implemented as a computationally simpler operation than sign extension.

If the address fault control flag 116 is set and the address space control flag 112 specifies that an application is to be confined to an address space subset, processor 100 can check (280) whether a generated address reference is not in the proper address space subset (e.g., page 6, lines 12-16; page 8, lines 6-8). For example, after a truncated address is sign or zero extended, the resultant address reference can be compared (285) to the original generated address reference (e.g., page 6, lines 16-18; page 8, lines 8-11). When the two are different, the processor 100 generates an address fault (290) (e.g., page 6, lines 18-19; page 8, lines 12-13). If the address

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

space control flag 112 does not specify that an application is to be confined to an address space subset, the processor 100 can ignore the address fault control flag 116 (*e.g.*, page 7, lines 6-12).

Independent claim 1 recites a processor 100. The processor 100 includes several mean-plus-function elements:

- A means for executing an instruction of an application of a first bit size ported to a second bit size environment, the second bit size being greater than the first bit size. The disclosed structure is the instruction execution core 130 of the processor 100 (*e.g.*, page 12, lines 5-6); and
- A means for confining the application to an address space subset of the first bit size. The disclosed structure is the instruction execution core 130 (*e.g.*, page 5, lines 8-9) and/or associated logic (*e.g.*, page 5, lines 8-9).

The “means for confining” comprises means elements:

- A means for truncating generated address references of the second bit size to the first bit size. The disclosed structure is the instruction execution core 130 and/or associated logic (*e.g.*, page 5, lines 8-9; page 12, lines 10-11).
- A means for determining that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control signal, the address format control signal having a first setting to indicate unsigned address space and a second setting to indicate signed address space. The disclosed structure is the instruction execution core (*e.g.*, page 12, lines 20-22), in conjunction with the address format control flag 114 (*e.g.*, page 5, line 21 to page 6, line 9); and
- A means for extending to the second bit size the truncated generated address references based on results from said means for determining, zero-extending the truncated generated address references if the address space subset of the first bit size is unsigned and sign-extending the truncated generated address references if the address space subset of the first bit size is signed. The disclosed structure is the instruction execution core (*e.g.*, page 5, line 21 to page 6, line 9, *with* page 12, lines 12-13).

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

Independent claim 7 recites a processor 100 comprising a memory 120 to store an instruction of an application ported from a first bit size environment to a second bit size environment, the second bit size being greater than the first bit size (*e.g.*, page 5, lines 3-5); and an instruction execution core 130 coupled to said memory 120 (*e.g.*, page 5, lines 5-6). The instruction execution core 130 is to execute the instruction of the application (*e.g.*, page 4, lines 11-12; page 7, lines 16-22). The instruction execution core 130 is to determine that the application is confined to an address space subset of the first bit size (*e.g.*, page 3, lines 4-7; page 4, lines 19-25; page 5, lines 1-4; page 7, lines 23-25; page 8, lines 17-19); generate an address reference of the second bit size as part of execution of the instruction (*e.g.*, page 7, line 25 to page 8, line 1; step 230 in FIG. 2); truncate the generated address reference from the second bit size to the first bit size (*e.g.*, page 3, lines 8-11; page 5, lines 1-3, 8-9; page 8, lines 1-2; step 240 in FIG. 2); determine that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control flag (*e.g.*, page 5, lines 21-22; page 8, lines 2-3; decision 250 in FIG. 2), the address format control flag having a first setting to indicate unsigned address space and a second setting to indicate signed address space (*e.g.*, page 2, lines 14-18); zero extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be unsigned address space (*e.g.*, page 5, lines 12-14, 22 to page 6, line 1; page 8, lines 5-6; step 270 in FIG. 2); and sign extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space (*e.g.*, page 5, lines 9-12; page 6, lines 1-7; page 8, lines 3-5; step 260 in FIG. 2).

Independent claim 15 recites a method comprising determining that an application is confined to an address subset of a first bit size, the application including an instruction (*e.g.*, page 3, lines 4-7; page 4, lines 19-25; page 5, lines 1-4; page 7, lines 23-25; page 8, lines 17-19); generating an address reference of a second bit size as part of execution of the instruction (*e.g.*, page 7, line 25 to page 8, line 1; step 230 in FIG. 2); truncating the generated address reference from the second bit size to the first bit size (*e.g.*, page 3, lines 8-11; page 5, lines 1-3, 8-9; page 8, lines 1-2; step 240 in FIG. 2); determining that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control flag (*e.g.*, page 5, lines 21-22; page 8, lines 2-3; decision 250 in FIG. 2), the address format control flag having a first setting to indicate unsigned address space and a second setting to



Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

indicate signed address space (*e.g.*, page 2, lines 14-18); zero-extending the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be unsigned address space (*e.g.*, page 5, lines 12-14, 22 to page 6, line 1; page 8, lines 5-6; step 270 in FIG. 2); and sign-extending the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space (*e.g.*, page 5, lines 9-12; page 6, lines 1-7; page 8, lines 3-5; step 260 in FIG. 2).

Dependent claim 11 recites the processor of claim 7, wherein the instruction execution core is to generate an address fault flag based at least in part on a comparison of the generated address reference and the extended, truncated, generated address reference (*e.g.*, page 6, lines 12-19).

Dependent claim 12 recites the processor of claim 11, wherein the instruction execution core is to generate the address fault flag only if: the comparison of the generated address reference and the extended, truncated, generated address reference indicates that the compared addresses are different, and an address fault control flag specifies to check for an address fault (*e.g.*, page 6, lines 12-19; page 7, lines 1-12; page 8, lines 6-13; steps 280, 285, and 290 in FIG. 2).

Dependent claim 27 recites the processor of claim 12, wherein the instruction execution core is to generate the address fault flag if: the application is confined to the address space subset of the first bit size, the comparison of the generated address reference and the extended, truncated, generated address reference indicates that the compared addresses are different, and the address fault control flag specifies to check for the address fault (*e.g.*, page 6, lines 12-19; page 7, lines 1-12; page 8, lines 6-13; steps 280, 285, and 290 in FIG. 2).

## **VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL**

The Final Office Action rejects claim 27 under 35 U.S.C. § 112, second paragraph; and rejects claims 1, 2, 4, 7-9, 11, 13-15, 17-18, 22, 25-26, and 28 under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,420,992 to Killian *et al.* ("Killian").

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

## VII. ARGUMENT

### A. Legal Background

“During patent examination, the pending claims must be given their broadest reasonable interpretation consistent with the specification.” MPEP § 2111. “The broadest reasonable interpretation of the claims must also be consistent with the interpretation that those skilled in the art would reach.” *Id.* “Ordinary, simple English words whose meaning is clear and unquestionable, absent any indication that their use in a particular context changes there meaning, are construed to mean exactly what they say.” MPEP § 2111.01.

“A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference.” MPEP § 2131 citing Verdegaal Bros. v. Union Oil Co. of California, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). “The identical invention must be shown in as complete detail as is contained in the ... claim.” MPEP § 2131 citing Richardson v. Suzuki Motor Co. Ltd., 9 USPQ2d 1913, 1920 (Fed. Cir. 1989) (ellipse in MPEP). Every element of the claimed invention must be literally present, arranged as in the claim. See Richardson, 9 USPQ2d at 1920.

### B. Argument

#### 1. Rejection Under § 112, Second Paragraph

The Action of October 28, 2005 on page 3 says “claim 12 states that a flag is generated only if A and B are true (A and B being symbolic of the actual limitations). Claim 27 then states that the flag is generated if A, B, and C are true. Does applicant want to generate a flag if A, B, and C are true and not only if A and B are true? If so, then this is not clear because the limitation of claim 12 must still hold. If not, then it is not clear why applicant introduces element C, when all that matters are elements A and B.”

Using the symbolic notation adopted in the Action, Claim 12 recites “the instruction execution core is to generate the address fault flag only if” A and B. In comparison, claim 27 recites “the instruction execution core is to generate the address fault flag if” A, B, and C.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

The plain meaning of “if” and “only if” are:

“P if Q” means “If Q then P”; and

“P only if Q” means “If P then Q”.

As explained in “Introduction to Symbolic Logic” by John L. Pollock (Holt, Rinehard, and Winston, 1969):

“P if Q” means the same thing as “If Q then P.” Consider the statement “The crops will be destroyed if there is a flood.” To say this is not to say that the crops might not be destroyed anyway, for example, by a drought. But the Statement “If the crops are destroyed there is a flood” precludes the possibility of them being destroyed by a drought, therefore it cannot be a proper paraphrase of “The crops will be destroyed if there is a flood.” The proper paraphrase is “If there is a flood then the crops will be destroyed.” ... “P only if Q” works just the other way around. It means “If P then Q.” Consider the statement “The crops will be destroyed only if there is a flood.” This means that the only way the crops can be destroyed is by a flood, and hence if the crops *are* destroyed then there must have been a flood. This then means “If the crops are destroyed then there is a flood.”

— Pollock page 17 (pages 9-19 of Pollock appear in the *Evidence Appendix*, and were previously submitted with the Response of January 30, 2006).

The statement in the Action on page 2 that “a flag is generated only if A and B” means “as long as A and B are true, then no matter what else happens, the flag is generated” is incorrect. Claim 12 describes that the address fault flag is generated only if A and B. A paraphrasing of the limitations of claim 12 would be “If the address fault flag is generated, then A and B must be true.” A and B must be true as a prerequisite to generating the address flag, but it is within the scope of claim 12 that A and B might be true without generation of an address fault flag.

Claim 27 describes that the address fault flag is generated if A, B, and C. A paraphrasing of the limitations of claim 27 would be “If A, B, and C are true, then the address fault flag is

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

generated.” Thus, claims 12 and 27 describe different logical propositions, such that claim 27 is further limiting claim 12.

Accordingly, appellants respectfully request that the § 112 rejection of claim 27 be reversed.

## 2. Rejection Under § 102(b)

The Examiner has not presented a case for anticipation based on Killian. The operations in Killian are not performed *as arranged* in the claims.

Referring to independent claim 7 as an example, claim 7 recites:

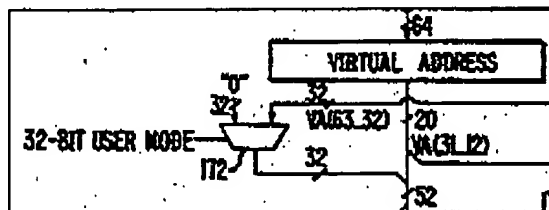
*truncate the generated address reference of a second bit size to the first bit size;*

*determine that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control flag, the address format control flag having a first setting to indicate unsigned address space and a second setting to indicate signed address space;*

*zero extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be unsigned address space; and*

*sign extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space.*

The rejections and argument have centered around the operations illustrated in FIG. 5D of Killian, as noted in paragraph 27 of the Action of October 28, 2005, and elaborated upon in the Advisory Action. The most relevant portion of FIG. 5D is excerpted below:



- *Killian does not disclose the recited sign extension of the truncated address*

The Examiner contends that the splitting up of the bits coming out of the box labeled “Virtual Address” constitutes the truncation step, and that the operation of multiplexer 172,

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

which selectively replaces bits 63 to 32 with zeros depending upon the "User Mode" constitutes the zero-extension step.

Adopting these portions of the Examiner's analysis for the purpose of argument, there still is no sign-extending of the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space.

The Examiner contends that "In a mode other than 32-bit user mode, the right input (the 32 original sign bits) are selected by the multiplexer and sign-extension is performed since the sign bits are appended to the address." See Advisory Action.

This simply is not what the claim describes, and misses the point of the claimed invention, which is to eliminate sign errors potentially carried by the high order bits (e.g., sign errors introduced to a 32-bit address by a 64-bit sign-extension operation). Adding bits 63 to 32 back onto the bits 31 to 0 is not *sign extension* of the truncated, generated address reference to the second bit size. Any sign information contained in the high-order bits was carried by them before truncation.

Moreover, if it could be assumed that bits 63 to 32 properly reflected sign-extension of bit 31 based on an earlier operation (as referred to by the Examiner), there would be no reason for the zero-extension operation performed by multiplexer 172, since bits 63 to 32 would already be zero in 32-bit user mode. As explained by Killian:

"A multiplexer 172 is interposed in the path and substitutes 0's for VA(63 ... 32) in 32-bit user mode. The zeroing of the high order bits insures that a valid user address (VA (31)=0) that resulted in two's complement overflow is in sign-extended form." Killian col. 19, lines 63-68.

- ***Killian does not disclose the recited determination that the address space is unsigned***

The above passage demonstrates a further reason why the claims are not anticipated: the zero-extension performed by multiplexer 172 in Killian is not performed because of a determination that the address space subset of the first bit-size (in this case, 32 bits) is an unsigned address space. Rather, the address space in 32-bit user mode is expressly signed, with bit 31 being the sign bit. While Killian discloses that there is an address error if bit 31 is

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

anything but zero (*see, e.g.,* FIG. 3A), it still is expressly disclosed as a sign bit. *See* col. 17, line 60 to col. 18, line 7; col. 19, lines 23-40; and the passage at col. 19, lines 63-68 reproduced above. If the 32-bit user mode in Killian used unsigned address space, there would not be a sign bit.

Accordingly, appellants respectfully request that the § 102(b) rejection of claims 1, 2, 4, 7, 8, 9, 11, 13, 14, 15, 17, 18, 22, 25, 26, and 28 be reversed.

### VIII. CONCLUSION

Appellants respectfully request that the Board of Patent Appeals and Interferences reverse the Examiner's decision rejecting claims 1, 2, 4, 7, 8, 9, 11, 13, 14, 15, 17, 18, 22, 25, 26, 27, and 28 and direct the Examiner to pass the case to issue.

The Commissioner is hereby authorized to charge the appeal brief fee and any additional fees which may be necessary for consideration of this Brief to Kenyon & Kenyon Deposit Account No. 11-0600.

Respectfully submitted,  
KENYON & KENYON LLP



David A. Klein  
Reg. No. 46,835

Dated: June 28, 2006

Kenyon & Kenyon LLP  
1500 K Street, N.W.  
Suite 700  
Washington, D.C. 20005  
Tel: (202) 220-4200  
Fax: (202) 220-4201

607095\_1.DOC

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

## APPENDIX

### IX. CLAIMS ON APPEAL

The claims in their current form are presented below:

1. A processor comprising:

means for executing an instruction of an application of a first bit size ported to a second bit size environment, the second bit size being greater than the first bit size; and

means for confining the application to an address space subset of the first bit size, said means for confining comprising:

means for truncating generated address references of the second bit size to the first bit size;

means for determining that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control signal, the address format control signal having a first setting to indicate unsigned address space and a second setting to indicate signed address space; and

means for extending to the second bit size the truncated generated address references based on results from said means for determining, zero-extending the truncated generated address references if the address space subset of the first bit size is unsigned and sign-extending the truncated generated address references if the address space subset of the first bit size is signed.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

2. The processor of claim 1, wherein the first bit size is 32-bit and the second bit size is 64-bit.
4. The processor of claim 1, wherein the means for confining includes means for generating an address fault.
7. A processor comprising:
  - a memory to store an instruction of an application ported from a first bit size environment to a second bit size environment, the second bit size being greater than the first bit size; and
  - an instruction execution core coupled to said memory, said instruction execution core to execute the instruction of the application, said instruction execution core to
    - determine that the application is confined to an address space subset of the first bit size;
    - generate an address reference of the second bit size as part of execution of the instruction;
    - truncate the generated address reference from the second bit size to the first bit size;
    - determine that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control flag, the address format control flag having a first setting to indicate unsigned address space and a second setting to indicate signed address space;
    - zero extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be unsigned address space; and
    - sign extend the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space.



Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

8. The processor of claim 7, wherein the application ported from a first bit size environment to a second bit size environment is an application ported from a 32-bit environment to a 64-bit environment.

9. The processor of claim 7, wherein the instruction execution core is to determine that the application is confined to the address space subset of the first bit size based at least in part on an address space control flag.

11. The processor of claim 7, wherein the instruction execution core is to generate an address fault flag based at least in part on a comparison of the generated address reference and the extended, truncated, generated address reference.

12. The processor of claim 11, wherein the instruction execution core is to generate the address fault flag only if:

the comparison of the generated address reference and the extended, truncated, generated address reference indicates that the compared addresses are different, and  
an address fault control flag specifies to check for an address fault.

13. The processor of claim 7, wherein said memory is a cache memory.

14. The processor of claim 7, wherein the processor is a 64-bit processor.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

15. A method comprising:

determining that an application is confined to an address subset of a first bit size, the application including an instruction;

generating an address reference of a second bit size as part of execution of the instruction;

truncating the generated address reference from the second bit size to the first bit size;

determining that the address space subset of the first bit size is signed address space or unsigned address space based on a setting of an address format control flag, the address format control flag having a first setting to indicate unsigned address space and a second setting to indicate signed address space;

zero-extending the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be unsigned address space; and

sign-extending the truncated, generated address reference to the second bit size if the address space subset of the first bit size is determined to be signed address space.

17. The method of claim 15, wherein the application is ported from a 32-bit environment to a 64-bit environment.

18. The method of claim 15, wherein said determining that the application is confined to the address subset of the first bit size is based at least in part on an address space control flag.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

22. The method of claim 15, further comprising:

generating an address fault flag based at least in part on a comparison of the generated address reference and the extended, truncated, generated address reference.

23. The method of claim 22, wherein generating the address fault flag is further based in part on an address fault control flag, wherein the address fault flag is generated only if:

the comparison of the generated address reference and the extended, truncated, generated address reference indicates that the compared addresses are different, and

an address fault control flag specifies to check for an address fault.

24. The processor of claim 4, wherein said means for generating the address fault comprises:

means for comparing a generated address reference as input into said means for truncating with an extended, truncated, generated address reference as output by said means for extending,

wherein said means for generating the address fault outputs that an address fault has occurred only if:

an output of said means for comparing indicates that the compared addresses are different, and

an address fault control signal specifies to check for the address fault.

25. The processor of claim 2, wherein the unsigned address space is 4 gigabytes and the signed address space is -2 gigabytes to +2 gigabytes.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

26. The processor of claim 8, wherein the unsigned address space is 4 gigabytes and the signed address space is -2 gigabytes to +2 gigabytes.

27. The processor of claim 12, wherein the instruction execution core is to generate the address fault flag if:

the application is confined to the address space subset of the first bit size,

the comparison of the generated address reference and the extended, truncated, generated address reference indicates that the compared addresses are different, and

the address fault control flag specifies to check for the address fault.

28. The method of claim 17, wherein the unsigned address space is 4 gigabytes and the signed address space is -2 gigabytes to +2 gigabytes.

Appeal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

**X. EVIDENCE APPENDIX**

“Introduction to Symbolic Logic” by John L. Pollock, pages 9-19 (Holt, Rinehard, and Winston, 1969).

Appcal Brief dated June 28, 2006  
Application No. 09/536,452

PATENT  
Attorney Docket No. 2207/8754

**XL RELATED PROCEEDINGS APPENDIX**

Per Section II above, there are no related proceedings to the present Appeal.

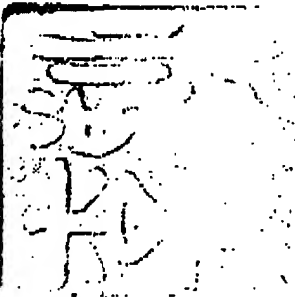
**EVIDENCE APPENDIX**

Steve Lee  
18 June 452  
271-9318

# INTRODUCTION TO SYMBOLIC LOGIC

JOHN L. POLLOCK  
*State University of New York  
at Buffalo*

HOLT, RINEHART and WINSTON, INC.  
New York Chicago San Francisco Atlanta  
Dallas Montreal Toronto London Sydney





*To Carrot, Klunky, Imbreilla,  
and Question Mark*

Copyright © 1969 by Holt, Rinehart and Winston, Inc.  
All rights reserved

Library of Congress Catalog Card Number: 69-12269

SBN: 03-072765-0

Printed in the United States of America

1 2 3 4 5 6 7 8 9

# II THE PROPOSITIONAL CALCULUS

## INTRODUCTION

8

Thus formal analyticity is another way to talk about sound argument forms. An argument form is sound if, and only if, its corresponding conditional is formally analytic. Thus, the argument form

All  $A$  are  $B$ .

All  $B$  are  $C$ .

Therefore, all  $A$  are  $C$ .

is sound, because the corresponding conditional "If all  $A$  are  $B$  and all  $B$  are  $C$ , then all  $A$  are  $C$ " is formally analytic—regardless of what we substitute for  $A$ ,  $B$ , and  $C$ , we get an analytic statement. This, then, is the most important connection between the two conceptions of logic. It gives us a way of talking about sound argument forms in terms of the formal analyticity of statements.

## II.1 THE SENTENTIAL CONNECTIVES

There is a large class of statements and arguments whose forms can be expressed adequately using the expressions "and," "or," "if... then," "if and only if," and "it is not the case that." These expressions are called the *sentential connectives*, because they are used to connect sentences to form larger sentences. For example, the statement "Either it is raining or it is not the case that it is raining" has the form " $P$  or it is not the case that  $P$ ," where now  $P$  stands for "It is raining." Similarly, the statement "If John goes to the party, then Joe will stay home" has the form "If  $P$  then  $Q$ ," where  $P$  stands for "John will go to the party" and  $Q$  stands for "Joe will stay home."

Unfortunately, in natural languages, such as English, statements that have the same form do not always look the same. For example, the statements "Either it is snowing or it is not the case that it is snowing" and "Either it is sleeting or it is not sleeting" have the same form; they are both formed by saying that either one statement is true or else a second statement is true, where the second statement simply denies the truth of the first statement. But they do not look quite the same because the "not" is placed differently in the two statements. It is therefore advantageous to introduce an artificial symbolism to represent the form of a statement. We replace "not" by " $\sim$ ," and then both "It is not the case that it is sleeting" and "It is not sleeting" can be represented by " $\sim$  (it is sleeting)."

Letting  $P$  and  $Q$  be arbitrary statements, we can introduce the following symbols:

9

## THE PROPOSITIONAL CALCULUS

Symbol	Meaning
$\sim P$	It is not the case that $P$ .
$(P \& Q)$	$P$ and $Q$ (or, both $P$ and $Q$ ).
$(P \vee Q)$	$P$ or $Q$ (or, either $P$ or $Q$ ).
$(P \supset Q)$	If $P$ then $Q$ .
$(P \equiv Q)$	$P$ if and only if $Q$ .

These symbols are abbreviations for the sentential connectives.<sup>1</sup> Statements formed using these symbols have names, as follows:  $\sim P$  is the *negation* of  $P$ ;  $(P \& Q)$  is the *conjunction* of  $P$  and  $Q$ ; and  $P$  and  $Q$  are the *conjuncts*;  $(P \vee Q)$  is the *disjunction* of  $P$  and  $Q$ ; and  $P$  and  $Q$  are the *disjuncts*;  $(P \supset Q)$  is a *conditional*, and  $P$  is the *antecedent*, and  $Q$  is the *consequent*;  $(P \equiv Q)$  is a *biconditional*.

Using these symbols we can represent the forms of quite complex statements. We have been using capital letters to symbolize the simple statements that form the parts of compound statements. We might occasionally need more than 26 letters, so we can also use capital letters with subscripts, such as  $P_7$  or  $Q_{10}$ . It should be emphasized that the same letter with different subscripts, such as  $P$  and  $P_7$ , can be used to represent totally unrelated statements. The fact that  $P$  occurs in  $P_7$  is just a coincidence. Capital letters, with or without subscripts, used this way are called *sentential letters*. In the statement "Either it is raining or it is not the case that it is raining," we can let  $P_0$  be the statement "It is raining." Then the above statement becomes "Either  $P_0$  or it is not the case that  $P_0$ ." Using the symbols for the sentential connectives, this in turn can be symbolized as "Either  $P_0$  or  $\sim P_0$ ," and then as  $(P_0 \vee \sim P_0)$ .

Now consider some more examples of symbolizing the forms of statements like "If he comes we will have the party at his house, and if he doesn't come then we will have the party at Jones' house." Letting  $P$  be the statement "He comes,"  $Q$  the statement "We will have the party at his house," and  $R$  the statement "We will have the party at Jones' house," we can first symbolize the statement as "If  $P$  then  $Q$ , and if it is not the case that  $P$  then  $R$ ." This in turn can be symbolized as "If  $P$  then  $Q$ , and if  $\sim P$  then  $R$ ," and then as  $(P \supset Q) \& (\sim P \supset R)$ , and finally as  $[(P \supset Q) \& (\sim P \supset R)]$ .

<sup>1</sup> In what follows, these symbols will be called the *sentential connectives*, although strictly speaking they are abbreviations for the sentential connectives. The sentential connectives themselves are English words and phrases.

## THE SENTENTIAL CONNECTIVES

As another example, consider "If Jones needs money, then either he will reduce prices or he will apply to the bank for a loan." Letting  $P$  be "Jones needs money,"  $Q$  be "He will reduce prices," and  $R$  be "He will apply to the bank for a loan," this statement can be symbolized, first as "If  $P$  then either  $Q$  or  $R$ ," then as "If  $P$  then  $(Q \vee R)$ ," and finally as  $[P \supset (Q \vee R)]$ .

There are two things that should be noticed about the preceding examples. The first is that in symbolizing relatively complex statements like "If he comes we will have the party at his house, and if he doesn't come then we will have the party at Jones' house," we begin by symbolizing the smallest parts, and then we construct the successively larger parts one at a time until we finally get the whole statement. The symbolization went as follows:

$$\begin{array}{l} \text{If } P \text{ then } Q, \text{ and if it is not the case that } P \text{ then } R \\ \hline (P \supset Q) \text{ and if } \quad \quad \quad \sim P \quad \quad \quad \text{then } R \\ \hline (P \supset Q) \text{ and } \quad \quad \quad (\sim P \supset R) \\ \hline [(P \supset Q) \& (\sim P \supset R)] \end{array}$$

It is always best to begin by symbolizing the smallest parts of a statement first, and then constructing the symbolization of the successively larger parts one at a time in terms of the parts that make them up. It is unwise to try to symbolize an entire statement in one fell swoop if the statement is at all complicated.

Second, notice the use of parentheses. Each time we symbolize a part of the statement—unless that part is a negation—we enclose it in parentheses to keep it separate from the other parts of the statement. The parentheses take the place of commas and other grammatical conventions of English. We must always be careful to put the parentheses in, or the resulting symbolization will be ambiguous. Consider the two statements "It is not true that Jones has a girlfriend and his wife is going to divorce him" and "It is not true that Jones has a girlfriend, and his wife is going to divorce him." These obviously mean quite different things. The first denies that it is true both that Jones has a girlfriend and that his wife is going to divorce him, whereas the second says that Jones does not have a girlfriend, but his

## 12 FORMULAS OF PROPOSITIONAL CALCULUS

13

precise rules for constructing formulas of the propositional calculus. The simplest formulas are simply sentential letters, with or without subscripts, such as  $P$ ,  $Q$ ,  $P_{13}$ ,  $R_{128}$ , and so forth. Let us call these *atomic formulas of the propositional calculus* because they are the atoms from which more complicated formulas are constructed. Then all other formulas of the propositional calculus can be constructed by successive applications of the following rules:

1. An atomic formula is a formula.
2. If  $P$  is any formula, then  $\sim P$  is a formula.
3. If  $P$  and  $Q$  are any formulas, then  $(P \& Q)$  is a formula.
4. If  $P$  and  $Q$  are any formulas, then  $(P \vee Q)$  is a formula.
5. If  $P$  and  $Q$  are any formulas, then  $(P \supset Q)$  is a formula.
6. If  $P$  and  $Q$  are any formulas, then  $(P \equiv Q)$  is a formula.

All formulas of the propositional calculus can be constructed by repeated application of these rules. Consider the formula

$$(P \supset (\sim Q \& R))$$

We can build this formula up using the above six rules as follows. We begin with the smallest parts and work outwards. By Rule 1,  $P$ ,  $Q$ , and  $R$  are formulas. Then by Rule 2,  $\sim Q$  is a formula. By Rule 3, as  $\sim Q$  and  $R$  are both formulas,  $(\sim Q \& R)$  is a formula. Then by Rule 5, as  $P$  and  $(\sim Q \& R)$  are both formulas,  $(P \supset (\sim Q \& R))$  is a formula.

We can construct very complicated formulas using these rules. For example, consider the formula

$$\begin{aligned} ((P \supset (Q_3 \equiv \sim R_4)) \\ \equiv \sim \sim (\sim P \& \sim (Q_3 \equiv (R_4 \vee (R_{17} \& \sim \sim P)))))) \end{aligned}$$

Let us see how we would build this formula up using the six rules. Again, we begin with the smallest parts and work outwards. By Rule 1,  $P$ ,  $Q_3$ ,  $R_4$ , and  $R_{17}$  are formulas, because they are atomic formulas. As  $P$  and  $R_4$  are formulas, we can use Rule 2 to construct the formulas  $\sim P$  and  $\sim R_4$ . As  $\sim P$  is then a formula, by Rule 2 again,  $\sim \sim P$  is a formula. Then as  $R_{17}$  and  $\sim \sim P$  are both formulas, by Rule 3,  $(R_{17} \& \sim \sim P)$  is a formula. Then as  $R_4$  and  $(R_{17} \& \sim \sim P)$  are both formulas, by Rule 4,  $(R_4 \vee (R_{17} \& \sim \sim P))$  is a formula. Then as  $Q_3$  and  $(R_4 \vee (R_{17} \& \sim \sim P))$  are both formulas, by Rule 6,

$$(Q_3 \equiv (R_4 \vee (R_{17} \& \sim \sim P)))$$

is a formula. Then by Rule 2,

$$\sim (Q_3 \equiv (R_4 \vee (R_{17} \& \sim \sim P)))$$

## 12 THE PROPOSITIONAL CALCULUS

wife is going to divorce him anyway. These two statements would be symbolized as  $\sim (P \& Q)$  and  $(\sim P \& Q)$  respectively. But if we omitted the parentheses in symbolizing these statements and just wrote  $\sim P \& Q$ , we would not know which of these two different statements were meant.

It is not necessary to enclose a negation (a statement of the form  $\sim P$ ) in parentheses, because " $\sim$ " does not really connect sentences—it acts on a single sentence. But for any of the other sentential connectives it is necessary to use parentheses each time the connective is used.

### 11.1 EXERCISES

Symbolize the forms of the following statements:

1. If it rains today the ground will be wet, and we will not be able to have a picnic.
2. It is not true that if it is cloudy then it will rain.
3. If the sun shines in the morning it will rain, and if the sun does not shine in the morning then it will not rain.
4. If we get plenty of sunshine, then if it rains the flowers will grow.
5. It is not the case that, there is a woman in the next room if and only if Jim said there is.
6. It is not the case that there is a woman in the next room, if and only if, Jim said there is.
7. Either the entrails will contain cockroaches or they will not, and if they do then the gods are angered, and if they do not then Venus will be in apposition to Jupiter.
8. If she is an acrobat or she is a clown, then she lives in that trailer.
9. Harry will go to the bank today if, and only if, the market drops, and if Harry goes to the bank today the supervisors will hide, and if the supervisors hide and Harry does not go to the bank today, then Emmett will lose his shirt on the stock market.
10. If Francis Bacon wrote *Hamlet* and Shakespeare wrote *Macbeth*, then either Shakespeare was Bacon, or the theater manager was a crook.

### 11.2 FORMULAS OF THE PROPOSITIONAL CALCULUS

Using sentential letters to stand for statements, and then combining them with sentential connectives, we can construct quite complex statement forms, called *formulas of the propositional calculus*. We can give

## FORMULAS OF PROPOSITIONAL CALCULUS

is *molecular* if it is not atomic. Then molecular formulas are those formulas that contain sentential connectives.

Sometimes we will want to talk about one formula being a *part* of another formula. This just means that the first occurs somewhere in the second. For example,  $P, Q, R, (P \supset Q), \sim(P \supset Q)$ , and

$$[\neg(P \supset Q) \vee R]$$

are all parts of the formula  $\sim [ \sim (P \supset Q) \vee R ]$ . Let us also adopt the convention of saying that a formula is part of itself. Then

$$\sim [\sim (P \supset Q) \vee R]$$

is also a part of  $\sim[\sim(P \supset Q) \vee R]$ . The *atomic parts* of a formula are those parts of the formula that are atomic. Thus  $P$ ,  $Q$ , and  $R$  are the atomic parts of  $\sim[\sim(P \supset Q) \vee R]$ , while  $(P \supset Q)$ ,  $\sim(P \supset Q)$ ,  $\sim[\sim(P \supset Q) \vee R]$  and  $\sim[\sim(P \supset Q) \vee R]$  are parts that are not atomic.

## II.2 EXERCISES

A. Show how the following formulas of the propositional calculus can be constructed using Rules 1 through 6. Diagram the construction as was done on page 14.

1.  $\langle\langle P \supset Q \rangle\rangle \equiv (\sim(Q \supset P) \vee \sim(Q \& \sim P))$
2.  $\sim(\langle\langle Q \& R \rangle\rangle \supset \sim(Q \& (R \supset (Q \vee \sim R))))$
3.  $\langle\langle P \vee (Q \& R) \rangle\rangle \equiv (\sim P \& (\sim Q \vee \sim R))$

B. Construct a table for each of the following expressions indicating: (1) whether it is a formula of the propositional calculus; (2) if it is a formula, whether it is atomic or molecular; (3) if it is a molecular formula, whether it is a negation, conjunction, disjunction, conditional or biconditional; (4) if it is a formula, what its atomic parts are; (5) if it is a formula, what its parts are:

1.  $P$
2.  $\sim P$
3.  $P \vee Q$
4.  $((P \vee Q) \supset \sim Q)$
5.  $(\sim(P \equiv Q))$
6.  $(P \vee Q \supset R)$
7.  $\sim((Q \& R) \equiv (\sim Q \vee \sim R))$
8.  $(P \vee P)$
9.  $(P \& \sim P)$
10.  $(P \supset Q) \equiv \sim(Q \supset P)$

14  
THE PROPOSITIONAL CALCULUS

is a formula. We have already seen that  $\sim P$  is a formula, so by Rule 3,

$$(\sim P \& \sim(Q_8 \equiv (R_4 \vee (R_{17} \& \sim P))))$$

is a formula. Then by Rule 2,

$$\sim(\sim P \& \sim(Q_3 \equiv (R_4 \vee (R_{11} \& (\sim \sim P))))))$$

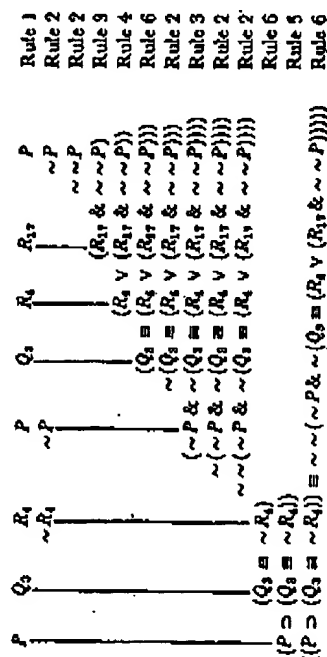
is a formula, and then by Rule 2 again,

$$\sim \sim (\sim P \& \sim (Q_3 \equiv \{R_4 \vee (R_{17} \& \sim \sim P)\})))$$

is a formula. This gives us the right side of the biconditional. Now working on the other side, as  $Q_2$  and  $\sim R_2$  are both formulas, by Rule 6,  $(Q_2 \supset \sim R_2)$  is a formula.  $P$  is a formula, so by Rule 5,  $(P \supset (Q_2 \supset \sim R_2))$  is a formula. Both sides of the biconditional have now been constructed, and then by Rule 6,

$$(((P \supset \{Q_3 \equiv \sim R_4\}) \\ = \sim \sim (\sim P \& \sim \{Q_3 \equiv (P_1 \vee (R_{17} \& \sim \sim P))\})))$$

is a formula. We can diagram the construction of this formula as follows:



Notice that whenever we construct a conjunction, disjunction, conditional, or biconditional, we enclose it in parentheses. In order to make formulas easier to read, parentheses are often replaced by brackets. Thus we might write the above formula as

$$[P \supset (Q_3 \equiv \sim R_4)] \\ \equiv \sim \sim [ \sim P \ \& \ \sim (Q_3 \equiv \{R_4 \vee (R_{17} \ \& \ \sim P)\}) ] ]$$

By doing this we can tell at a glance which left parenthesis or bracket goes with which right parenthesis or bracket; thus, we can see more easily what the formula means.

**Atomic formulas are the simplest formulas. Let us say that a formula**

## II.3 PARAPHRASING

Sometimes it is necessary to paraphrase a statement before it can be symbolized. Consider the statement "John and Joe both came to the party." We want to symbolize this as a conjunction, but it cannot be symbolized directly because "and" stands between two names rather than between two sentences. Before we can symbolize it we must paraphrase it so that the sentential connective connects two sentences, "John came to the party and Joe came to the party." Then letting  $P$  be "John came to the party" and  $Q$  be "Joe came to the party," we can symbolize it as  $(P \& Q)$ .

Another example of such paraphrasing is found in the following statement: "If either Kennedy or Khrushchev had been weaker willed concerning either Berlin or Cuba; the cold war would have turned into a hot war." This must be paraphrased first as "If either Kennedy had been weaker willed concerning either Berlin or Cuba, or Khrushchev had been weaker willed concerning either Berlin or Cuba, then the cold war would have turned into a hot war." Then this must be paraphrased again as "If either Kennedy had been weaker willed concerning Berlin or Kennedy had been weaker willed concerning Cuba, or Khrushchev had been weaker willed concerning Berlin or Khrushchev had been weaker willed concerning Cuba, then the cold war would have turned into a hot war." Finally then, this can be symbolized as  $[(P \vee Q) \vee (R \vee S)] \supset T$ .

Other kinds of paraphrasing may also be necessary. There are expressions in English such as "unless," "but," "if," "only if," "neither...nor," that are much like the sentential connectives. Whenever these occur in a statement, the statement must be paraphrased to replace them by sentential connectives. For example, "Neither Joe came to the party nor John came to the party" means the same thing as "Joe didn't come to the party and John didn't come to the party," and so it must be paraphrased in that way. In general, "Neither  $P$  nor  $Q$ " can be paraphrased as  $(\sim P \& \sim Q)$ .

"But" is like a forceful "and." We may say "He came but he didn't like it" rather than "He came and he didn't like it" merely to emphasize the second conjunct. This emphasis makes no difference for logic, so we can symbolize " $P$  but  $Q$ " as simply  $(P \& Q)$ .

The behavior of the expressions "if" and "only if" is somewhat surprising. There seems to be a strong temptation to identify " $P$  if  $Q$ "

## PARAPHRASING

with "If  $P$  then  $Q$ ," but in fact it should be the other way around; that is, " $P$  if  $Q$ " means the same thing as "If  $Q$  then  $P$ ." Consider the statement "The crops will be destroyed if there is a flood." To say this is not to say that the crops might not be destroyed anyway, for example, by a drought. But the statement "If the crops are destroyed there is a flood" precludes the possibility of them being destroyed by a drought, therefore it cannot be a proper paraphrase of "The crops will be destroyed if there is a flood." The proper paraphrase is "If there is a flood then the crops will be destroyed." In general, " $P$  if  $Q$ " can be symbolized as  $(Q \supset P)$ .

" $P$  only if  $Q$ " works just the other way around. It means "If  $P$  then  $Q$ ." Consider the statement "The crops will be destroyed only if there is a flood." This means that the only way the crops can be destroyed is by a flood, and hence if the crops are destroyed then there must have been a flood. This then means "If the crops are destroyed then there is a flood."

Note that " $P$  if and only if  $Q$ " is just the conjunction of " $P$  if  $Q$ " and " $P$  only if  $Q$ ," and thus that  $(P \equiv Q)$  is equivalent to

$$[(P \supset Q) \& (Q \supset P)].$$

One further expression that can be paraphrased in terms of the sentential connectives is "unless." " $P$  unless  $Q$ " can be paraphrased as  $(\sim Q \supset P)$ . Suppose we want to paraphrase the statement "We will go to the beach unless it rains." This is the same thing as saying "If it doesn't rain then we will go to the beach;" that is  $(\sim Q \supset P)$ .

## II.3 EXERCISES

Symbolize the forms of the following statements:

1. Neither Jack nor Jim will come unless Mary comes.
2. We will not get there on time unless we speed, but if we speed we might not get there at all.
3. We can get the door open only if we use an acetylene torch on it, but then the door will be ruined.
4. The river will not overflow its banks unless we either have an early thaw or heavy rains, but we will not have heavy rains.
5. Unless we have a flat tire, we can get there on time if we speed, but we will have a flat tire if we speed.

## THE PROPOSITIONAL CALCULUS

6. Neither Jack nor Jim will come if Mary comes, unless Joan and Mary both come.
7. Jeremy will get a Mercedes Benz for Christmas only if he does not offend Santa Claus, but Jeremy will offend Santa Claus if he does not believe in him, and Jeremy does not believe in Santa Claus.
8. Rain is imminent.
9. John will not come unless Jim comes, and Jim will not come if Jeffrey comes, but Jeffrey will only come if John does not come.
10. It will rain if the barometer drops, but if it rains it will cool off later, and it will not cool off later.

## II.4 DIVERGENT USES

One thing to beware of in symbolizing the forms of statements is that there are divergent uses of some of the sentential connectives in which they do not have their ordinary meaning. For example, "and" sometimes means "and then" rather than simply "and." In its ordinary use, "P and Q" means "It is true that P, and it is true that Q." On this reading, "P and Q" means the same thing as "Q and P." For example, "This is red and that is white" means the same thing as "That is white and this is red." But consider the use of "and" in the following sentence: "She got married and had a baby." This clearly does not mean the same thing as "She had a baby and got married." The former means "She got married and then had a baby." This use of "and" cannot be symbolized simply as "&." The same thing is true of "and" in "He lay down and fell asleep."

The sentential connective that has the greatest number of divergent uses is "if... then." What will be called the standard use of "if... then" is that in which "If P then Q" is true under the same conditions as "Either  $\sim P$  or Q"; that is,  $(\sim P \vee Q)$ . It is only this meaning of "If P then Q" that can be symbolized as  $(P \supset Q)$ . First it should be seen that "if... then" is at least sometimes used in this way. Consider the statement "If it rains then the crops will grow." Let us symbolize this as "If P then Q," where P means "It will rain" and Q means "The crops will grow." Now it will be argued that this is true under exactly the same circumstances as "Either it won't rain, or the crops will grow"; that is,  $(\sim P \vee Q)$ :

1. Suppose that "If P then Q" is true. So if it rains (if P is true), then Q is true. If it does not rain, then  $\sim P$  is true. But either it will

## DIVERGENT USES

not rain or it will rain. So either it will not rain, and then  $\sim P$  will be true, or else it will rain, and then Q will be true. Thus, no matter what happens, either  $\sim P$  will be true or Q will be true; that is,  $(\sim P \vee Q)$  will be true. So we see that if the statement "If P then Q" is true, then  $(\sim P \vee Q)$  is true.

2. In order to show that "If P then Q" and  $(\sim P \vee Q)$  are true under exactly the same circumstances, the converse of the above must now be proven, namely, that if  $(\sim P \vee Q)$  is true then "If P then Q" is true. So let us suppose that  $(\sim P \vee Q)$  is true. A disjunction "A or B" is true if, and only if, at least one of its disjuncts is true; that is, if, and only if, either A is true or B is true. Thus if  $(\sim P \vee Q)$  is true, then either  $\sim P$  is true or Q is true. Now it will be shown that "If P then Q" is true. If P is true, then  $\sim P$  cannot be true. But either  $\sim P$  or Q must be true. So if P is true then Q must be true; that is, "If P then Q" is true. Thus it has been shown that if  $(\sim P \vee Q)$  is true, then "If P then Q" is true.

Hence the two statements, "If it rains the crops will grow," and "Either it won't rain or the crops will grow," are true in exactly the same circumstances. In this context "if... then" has what is called its "standard use."

However, there are other uses of "if... then" in which "if P then Q" is clearly not the same as "Either  $\sim P$  or Q." One place this occurs is in what are called *counterfactual conditionals*. These are statements such as "If this match had been struck it would have lit," which tell us that if something that did not happen *had* happened, then something else would have been the case. To see that these conditionals cannot be translated into disjunctions, let us suppose that we have a match which was not struck. Then exactly one of the following two statements is true: "If this match had been struck it would have lit"; "If this match had been struck it wouldn't have lit." If "if... then" had its standard use in these statements, they would mean "Either this match wasn't struck or else it lit," and "Either this match wasn't struck or else it didn't light," respectively. But since the match was not struck, *both* of these disjunctions are true (because "This match wasn't struck" is true). Therefore, these disjunctions cannot mean the same thing as the counterfactual conditionals, because it is impossible for both of the counterfactual conditionals to be true at the same time. Either "If this match had been struck it would have lit" or "If this match had been struck it wouldn't have lit" is true, but they cannot both be true.

PTO/SB/97 (09-04)

Approved for use through 07/31/2006. OMB 0851-0031

U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE


Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

**Certificate of Transmission under 37 CFR 1.8**

I hereby certify that this correspondence for Application  
Serial No. 09/536,452 is being facsimile transmitted to the  
United States Patent and Trademark Office at 571-273-8300

on June 28, 2006.

Date



Signature

David A. Klein

Typed or printed name of person signing Certificate

46,835

Registration Number if applicable

202.220.4200

Telephone Number

**Note:** Each paper must have its own certificate of transmission, or this  
certificate must identify each submitted paper.

Transmittal Form (1 page)

Appeal Brief (20 pages)

Evidence Appendix (9 pages)

This collection of information is required by 37 CFR 1.8. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 1.8 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

*If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.*

DC01 618324 v1



**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**